A Multilevel Approach For Nonnegative Matrix Factorization

Nicolas Gillis[1] and François Glineur[1]

Abstract

Nonnegative Matrix Factorization (NMF) is the problem of approximating a nonnegative matrix with the product of two low-rank nonnegative matrices and has been shown to be particularly useful in many applications, e.g., in text mining, image processing, computational biology, etc. In this paper, we explain how algorithms for NMF can be embedded into the framework of multilevel methods in order to accelerate their convergence. This technique can be applied in situations where data admit a good approximate representation in a lower dimensional space through linear transformations preserving nonnegativity. A simple multilevel strategy is described and is experimentally shown to speed up significantly three popular NMF algorithms (alternating nonnegative least squares, multiplicative updates and hierarchical alternating least squares) on several standard image datasets.

Keywords: nonnegative matrix factorization, algorithms, multigrid and multilevel methods, image processing.

1 Introduction

Nonnegative Matrix Factorization (NMF) consists in approximating a nonnegative matrix as the product of two low-rank nonnegative matrices [27, 23]. More precisely, given a nonnegative matrix M of dimensions $m \times n$ and a factorization rank r, we would like to find two nonnegative matrices V and W with dimensions $m \times r$ and $r \times n$ such that

$$M \approx VW.$$

This decomposition can be interpreted as follows: denoting by $M_{:j}$ the j^{th} column of M, by $V_{:k}$ the k^{th} column of V and by W_{kj} the entry of W located at position (k, j), we want

$$M_{:j} \approx \sum_{k=1}^{r} W_{kj} V_{:k}, \quad W_{kj} \geq 0, \quad 1 \leq j \leq n,$$

so that each given (nonnegative) vector $M_{:j}$ is approximated by a nonnegative linear combination of r basis elements $V_{:k}$ to be found. Nonnegatity of vectors $V_{:k}$ ensures that these basis elements belong to the same space $\mathbb{R}^m_+$ as the columns of M and can then be interpreted in the same way. Moreover, the additive reconstruction due to nonnegativity of coefficients W_{kj} leads to a *part-based representation* [23]: basis elements $V_{:k}$ will tend to represent common parts of the columns of M. For example, let each column of M be a vectorized gray-level image of a face using (nonnegative) pixel intensities. The nonnegative matrix factorization of M will generate a matrix V whose columns are nonnegative basis elements of the original images which can then be interpreted as images as well. Moreover, since each

[1]Université catholique de Louvain, CORE, B-1348 Louvain-la-Neuve, Belgium. E-mail: nicolas.gillis@uclouvain.be and francois.glineur@uclouvain.be. Nicolas Gillis is a research fellow of the Fonds de la Recherche Scientifique (F.R.S.-FNRS). This text presents research results of the Belgian Program on Interuniversity Poles of Attraction initiated by the Belgian State, Prime Minister's Office, Science Policy Programming. The scientific responsibility is assumed by the authors.

original face is reconstructed through a weighted sum of these basis elements, the latter are common parts extracted from the original faces, such as eyes, noses and lips. Figure 1 illustrates this property of the NMF decomposition.

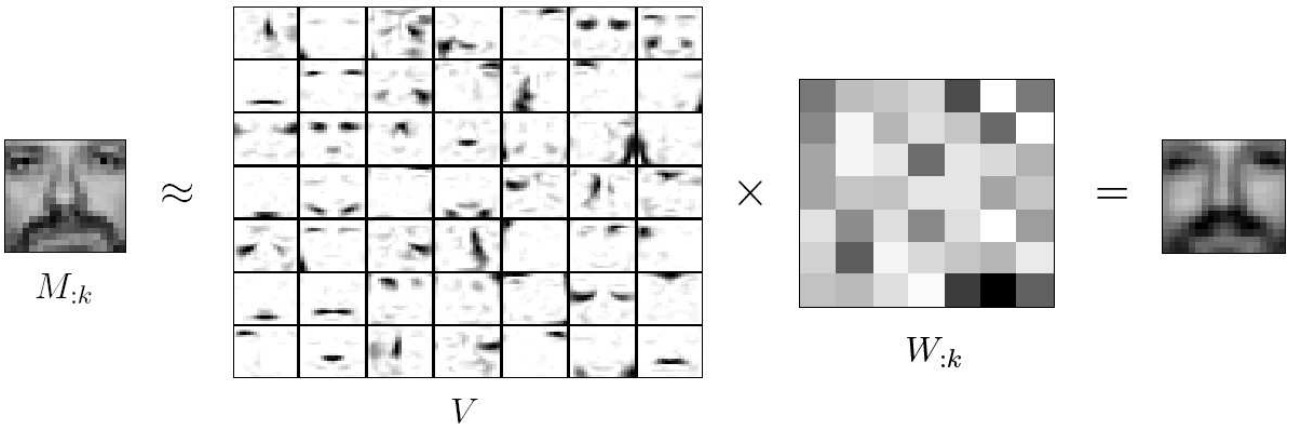


Figure 1: Illustration of NMF. Basis elements (matrix V) obtained with NMF on the CBCL Face Database #1, MIT Center For Biological and Computation Learning, available at *http://cbcl.mit.edu/cbcl/software-datasets/FaceData2.html*, consisting of 2429 gray-level images of faces (columns) with 19×19 pixels (rows) for which we set $r = 49$.

One of the main challenges of NMF is to design fast and efficient algorithms generating the nonnegative factors. In fact, on the one hand, practitioners need to compute rapidly good factorizations for large-scale problems (e.g., in text mining or image processing); on the other hand, NMF is a NP-hard problem [33] and we cannot expect to find a globally optimal solution in a reasonable computational time. This paper presents a general framework based on a multilevel strategy leading to faster convergence of NMF algorithms when dealing with data admitting a simple approximate low-dimensional representation (using linear transformations preserving nonnegativity), such as images. In fact, in these situations, a hierarchy of lower-dimensional problems can be constructed and used to compute efficiently approximate solutions of the original problem. Similar techniques have already been used for other dimensionality reduction tasks such as PCA [29].

The paper is organized as follows: NMF is first formulated as an optimization problem and three well-known algorithms (ANLS, MU and HALS) are briefly presented. We then introduce the concept of multigrid/multilevel methods and show how and why it can be used to speed up NMF algorithms. Finally, we experimentally demonstrate the usefulness of this approach on several standard image databases.

2 Algorithms for NMF

NMF is typically formulated as a nonlinear optimization problem with an objective function measuring the quality of the low-rank approximation. In this paper, we consider the sum of squared errors:

$$\min_{\substack{V \in \mathbb{R}^{m \times r} \\ W \in \mathbb{R}^{r \times n}}} ||M - VW||_F^2 \quad \text{s.t.} \quad V \geq 0,\ W \geq 0, \tag{NMF}$$

i.e., use the squared Frobenius norm $||A||_F^2 = \sum_{i,j} A_{ij}^2$ of the approximation error. Since (NMF) is NP-hard [33], most NMF algorithms focus on finding locally optimal solutions. In general, only convergence to stationary points of (NMF) (points satisfying the necessary first-order optimality conditions) is guaranteed.

2.1 Alternating Nonnegative Least Squares (ANLS)

Although (NMF) is a nonconvex problem, it is convex separately in each of the two factors V and W, i.e., finding the optimal factor V corresponding to a fixed factor W reduces to a convex optimization problem, and vice-versa. More precisely, this convex problem corresponds to a nonnegative least squares (NNLS) problem, i.e., a least squares problem with nonnegativity constraints. The so-called alternating nonnegative least squares (ANLS) algorithm for (NMF) minimizes (exactly) the cost function alternatively over factors V and W so that a stationary point of (NMF) is obtained in the limit [21]. A frequent strategy to solve the NNLS subproblems is to use active set methods [22] (see

Algorithm 1 Alternating Nonnegative Least Squares

Require: Data matrix $M \in \mathbb{R}^{m \times n}_+$ and initial iterate $W \in \mathbb{R}^{r \times n}_+$.

1: **while** stopping criterion not met **do**
2: $V \leftarrow \text{argmin}_{V \geq 0} ||M - VW||_F^2$;
3: $W \leftarrow \text{argmin}_{W \geq 0} ||M - VW||_F^2$.
4: **end while**

Appendix A) for which an efficient implementation[1] is described in [32, 21]. We refer the reader to [6] for a survey about NNLS methods.

2.2 Multiplicative Updates (MU)

In [24] Lee and Seung propose multiplicative updates (MU) for (NMF) which guarantee nonincreasingness of the objective function (cf. Algorithm 2). They also alternatively update V for W fixed and vice versa, using an technique which was originally proposed by Daube-Witherspoon and Muehllehner to solve nonnegative least squares problems [13]. The popularity of this algorithm came along with

Algorithm 2 Multiplicative Updates

Require: Data matrix $M \in \mathbb{R}^{m \times n}_+$ and initial iterates $(V, W) \in \mathbb{R}^{m \times r}_+ \times \mathbb{R}^{r \times n}_+$.

1: **while** stopping criterion not met **do**
2: $V \leftarrow V \circ \frac{[MW^T]}{[V(WW^T)]}$;
3: $W \leftarrow W \circ \frac{[V^T M]}{[(V^T V)W]}$.
4: **end while**

$\frac{[.]}{[.]}$ denotes the Hadamard (component-wise) division.

the popularity of NMF. Algorithm 2 does not guarantee convergence to a stationary point (although it can be slightly modified in order to get this property [25, 15]) and it has been observed to converge relatively very slowly, see [19] and references therein.

2.3 Hierarchical Alternating Least Squares (HALS)

In ANLS, variables are partitioned at each iteration such that each subproblem is convex. However, the resolution of these convex NNLS subproblems is nontrivial and relatively expensive. If we optimize instead one single variable at a time, we get a simple univariate quadratic problem which admits a closed-form solution. Moreover, since the optimal value of each entry of V (resp. W) does not depend of the other entries of the same column (resp. row), one can optimize alternatively whole columns of V and whole rows of W. This method was first proposed by Cichocki et al. [10, 8] and independently by

[1] Available at `http://www.cc.gatech.edu/~hpark/`.

[20, 16], and is herein referred to as Hierarchical Alternating Least Squares (HALS), see Algorithm 3. Under some mild assumptions, every limit point is a stationary point of (NMF) [20].

Algorithm 3 Hierarchical Alternating Least Squares

Require: Data $M \in \mathbb{R}_+^{m\times n}$ and initial iterates $(V, W) \in \mathbb{R}_+^{m\times r} \times \mathbb{R}_+^{r\times n}$.

while stopping criterion not met **do**
  Compute $A = MW^T$ and $B = WW^T$.
  for $k = 1 : r$ **do**
    $V_{:k} \leftarrow \max\left(0, \frac{A_{:k} - \sum_{l=1, l\neq k}^{r} V_{:l} B_{lk}}{B_{kk}}\right)$;
  end for
  Compute $C = V^T M$ and $D = V^T V$.
  for $k = 1 : r$ **do**
    $W_{k:} \leftarrow \max\left(0, \frac{C_{k:} - \sum_{l=1, l\neq k}^{r} D_{kl} W_{l:}}{D_{kk}}\right)$;
  end for
end while

3 Multigrid Methods

In this section, we briefly introduce multigrid methods. The aim is to give the reader some insight on these techniques in order to comprehend their applications for NMF. We refer the reader to [3, 4, 5, 31] and references therein for detailed discussions on the subject.

Multigrid methods were initially used to develop fast numerical solvers for boundary value problems. Given a differential equation on a continuous domain with boundary conditions, the aim is to find an approximation of a *smooth* function f satisfying the constraints. In general, the first step is to discretize the continuous domain, i.e., choose a set of points (a *grid*) where the function values will be approximated. Then, a numerical method (e.g., finite differences, finite elements) translates the continuous problem into a (square) system of linear equations:

$$\text{find } x \in \mathbb{R}^n \quad \text{s.t.} \quad Ax = b, \quad \text{with } A \in \mathbb{R}^{n\times n},\ b \in \mathbb{R}^n, \tag{3.1}$$

where the vector x will contain the approximate values of f on the grid points. Linear system (3.1) can be solved either by direct methods (e.g., Gaussian elimination) or iterative methods (e.g., Jacobi and Gauss-Seidel iterations). Of course, the computational cost of these methods depends on the number of points in the grid, which leads to a trade-off between precision (number of points used for the discretization) and computational cost.

Iterative methods update the solution at each step and hopefully converge to a solution of (3.1). Here comes the utility of multigrid: instead of working on a fine grid during all iterations, the solution is initially *restricted* to a coarser grid on which the iterations are cheaper. Moreover, the smoothness of function f allows to recover its low-frequency components faster on coarser grids. Solutions of the coarse grid are then *prolongated* to the finer grid and iterations can continue (higher frequency components of the error are reduced faster). Because the initial guess generated on the coarser grid is (hopefully) a good approximation of the final solution, less iterations are needed on the fine (expensive) grid to converge. Essentially, multigrid methods make iterative methods more efficient, i.e., accurate solutions are obtained faster.

More recently, these same ideas have been applied to a broader class of problems, e.g., multiscale optimization with trust-region methods [18] and multiresolution techniques in image processing [30].

4 Multilevel Approach for NMF

The three algorithms presented in Section 2 (ANLS, MU and HALS) are iteratively trying to find a stationary point of (NMF). Actually, most practical NMF algorithms are *iterative methods*, such as projected gradient methods [26], Newton-like methods [9, 14], ... (see also [1, 7, 11, 20] and references therein). In order to embed these algorithms in a multilevel strategy, one has to define the different levels and describe how the variables and the data are transfered between them. In this section, a general description of the multilevel approach for NMF algorithms is first presented and then applied on image datasets.

4.1 Description

Let each column of the matrix M be a element of the dataset (e.g., a vectorized image) belonging to $\mathbb{R}^m_+$. We define the restriction operator $\mathcal{R}$ as a linear operator

$$\mathcal{R} : \mathbb{R}^m_+ \to \mathbb{R}^{m'}_+ : x \to \mathcal{R}(x) = Rx,$$

with $R \in \mathbb{R}^{m' \times m}_+$ and $m' < m$; and the prolongation $\mathcal{P}$ as a linear operator

$$\mathcal{P} : \mathbb{R}^{m'}_+ \to \mathbb{R}^m_+ : y \to \mathcal{P}(y) = Py,$$

with $P \in \mathbb{R}^{m \times m'}_+$. Nonnegativity of matrices R and P is a sufficient condition to preserve nonnegativity of the solutions when they are transfered from one level to another. In fact, in order to generate nonnegative solutions, one requires

$$\mathcal{R}(x) \geq 0, \forall x \geq 0 \text{ and } \mathcal{P}(y) \geq 0, \forall y \geq 0.$$

We also define the corresponding transfer operators on matrices, operating columnwise:

$$\mathcal{R}([x_1\, x_2 \ldots x_n]) = [\mathcal{R}(x_1)\, \mathcal{R}(x_2) \ldots \mathcal{R}(x_n)], \text{ and}$$

$$\mathcal{P}([y_1\, y_2 \ldots y_n]) = [\mathcal{P}(y_1)\, \mathcal{P}(y_2) \ldots \mathcal{P}(y_n)],$$

for $x_i \in \mathbb{R}^m_+, y_i \in \mathbb{R}^{m'}_+, 1 \leq i \leq n$.

In order for the multilevel strategy to work, the information lost when transferring from one level to another must be limited, i.e., the data matrix M has to be well represented by $\mathcal{R}(M)$ in the lower dimensional space, which means that the reconstruction $\mathcal{P}(\mathcal{R}(M))$ must be close to M. From now on, we say that M is smooth with respect to $\mathcal{R}$ and $\mathcal{P}$ if and only if

$$s_M = \frac{||M - \mathcal{P}(\mathcal{R}(M))||_F}{||M||_F} \quad \text{is small .}$$

s_M gives a measure of how well M can be mapped by $\mathcal{R}$ into a lower-dimensional space and then brought back by $\mathcal{P}$, and still be a fairly good approximation of itself.

Based on these definitions, elaborating a multilevel approach for NMF is straightforward:

1. We are given $M \in \mathbb{R}^{m \times n}_+$ and $(V_0, W_0) \in \mathbb{R}^{m \times r}_+ \times \mathbb{R}^{r \times n}_+$;

2. Compute $M' = \mathcal{R}(M) = RM \in \mathbb{R}_+^{m' \times n}$ and $V_0' = \mathcal{R}(V_0) = RV_0 \in \mathbb{R}_+^{m' \times r}$, i.e., restrict the elements of your dataset and the basis elements of the current solution to a lower dimensional space;

3. Compute a rank-r NMF (V', W) of M' using (V_0', W_0) as initial matrices, i.e.,

$$V'W \approx M' = \mathcal{R}(M).$$

This can be done using any NMF iterative algorithm or, even better, using the multilevel strategy recursively (cf. Section 4.3).

4. Since

$$M \approx \mathcal{P}(\mathcal{R}(M)) = \mathcal{P}(M') \approx \mathcal{P}(V'W) = PV'W = \mathcal{P}(V')W = VW,$$

where V is the prolongation of V', (V, W) is a good initial estimate for a rank-r NMF of M, *provided that* M is smooth with respect to $\mathcal{R}$ and $\mathcal{P}$ (i.e., s_M is small) and that $V'W$ is a good approximation of $M' = \mathcal{R}(M)$ (i.e., $||M' - V'W||_F$ is small); in fact,

$$\begin{aligned} ||M - \mathcal{P}(V')W||_F &\leq ||M - \mathcal{P}(\mathcal{R}(M))||_F + ||\mathcal{P}(\mathcal{R}(M)) - \mathcal{P}(V'W)||_F \\ &\leq s_M||M||_F + ||\mathcal{P}(\mathcal{R}(M) - V'W)||_F \\ &\leq s_M||M||_F + ||P||_F\,||\mathcal{R}(M) - V'W||_F. \end{aligned}$$

5. Further improve the solution (V, W) using any NMF iterative algorithm.

Because computations needed at step 3 are cheap since $m' < m$, and in addition the low-frequency components of the error[2] is reduced faster on coarse levels (cf. Section 4.4), this strategy will accelerate the convergence of NMF algorithms.

We now illustrate this technique on image datasets, more precisely, on two-dimensional gray-level images. In general, images are composed of several smooth components, i.e., regions where pixel values are similar and change continuously with respect to their location (e.g., skin on a face or, the pupil or sclera[3] of an eye), that is, a pixel value can often be approximated using the pixel values of its neighbors. This observation can be used to define the transfer operators (Section 4.2). For the computation of a NMF solution needed at step 3, the multilevel approach can be used recursively; three strategies (called multigrid cycles) are described in Section 4.3. Finally, numerical results are reported in Section 5.

4.2 Coarse Grid and Transfer Operators

A crucial step of multilevel methods is to define the different levels and the transformations (operators) between them. Figure 2 is an illustration of a standard *coarse grid* definition: we note I^1 the matrix of dimension $(2^a+1) \times (2^b+1)$ representing the initial image and I^l the matrix of dimension $(2^{a-l+1}+1) \times (2^{b-l+1}+1)$ representing the image at level l obtained by keeping, in each direction, only one out of every two points of the grid at the preceding level, i.e., I^{l-1}.

The transfer operators describe how to transform the images when going from finer to coarser levels, and vice versa, i.e., how to compute the values (pixel intensities) of the image I^l using values from image I^{l-1} at the finer level (restriction) or from image I^{l+1} at the coarser level (prolongation). For the *restriction*, the *full-weighting* operator is a standard choice: values of the coarse grid points are the weighted average of the values of their neighbors on the fine grid (see Figure 3 for an illustration). Noting $I^l_{i,j}$ the intensity of the pixel (i, j) of image I^l with $0 \leq i \leq 2^a$ and $0 \leq j \leq 2^b$, it is defined as

[2]The low-frequency components refers to the parts of the data which are well-represented on coarse levels.

[3]The white part of the eye.

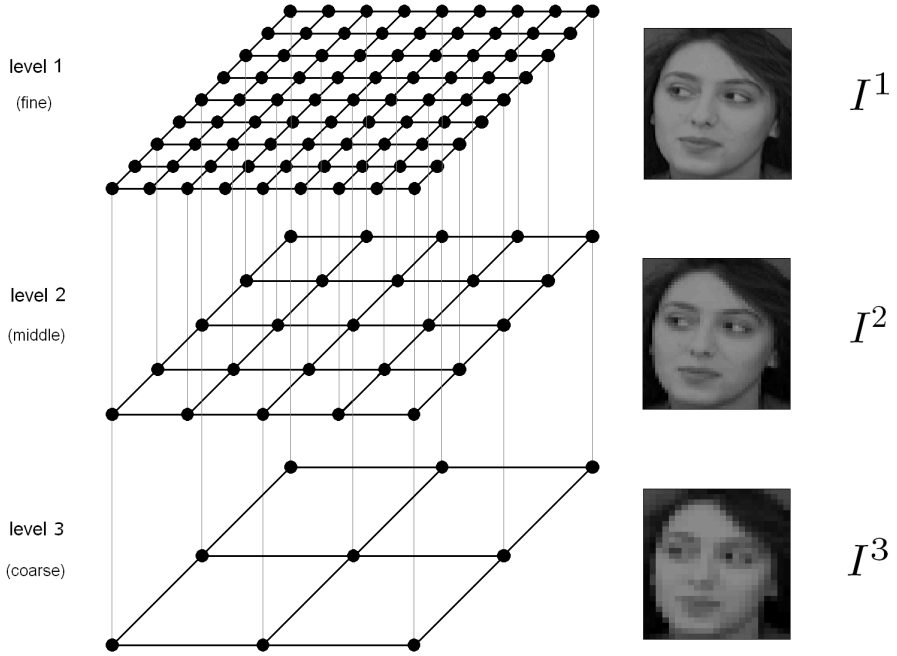


Figure 2: Multigrid Hierarchy. Schematic view of a grid definition for image processing (image from ORL face database, cf. Section 5).

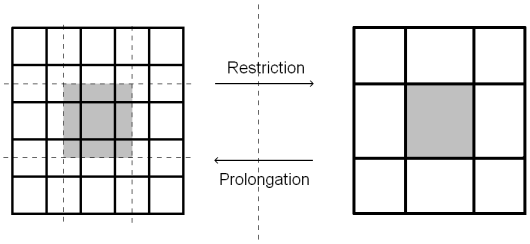


Figure 3: Restriction and Prolongation.

follows:

$$\begin{aligned} I^{l+1}_{i,j} = &\frac{1}{16}\Big[I^l_{2i-1,2j-1} + I^l_{2i-1,2j+1} + I^l_{2i+1,2j-1} + I^l_{2i+1,2j+1} \\ &+ 2(I^l_{2i,2j-1} + I^l_{2i-1,2j} + I^l_{2i+1,2j} + I^l_{2i,2j+1}) \\ &+ 4I^l_{2i,2j}\Big], \end{aligned} \tag{4.1}$$

except on the boundaries of the image (when $i = 0$, $j = 0$, $i = 2^a$ and/or $j = 2^b$) where the weights are adapted correspondingly. For example, to restrict a 3×3 image to a 2×2, $\mathcal{R}$ is defined with

$$R = \frac{1}{9}\begin{pmatrix} 4 & 2 & 0 & 2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 2 & 4 & 0 & 1 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 1 & 0 & 4 & 2 & 0 \\ 0 & 0 & 0 & 0 & 1 & 2 & 0 & 2 & 4 \end{pmatrix},$$

(3×3 images needing first to be vectorized to vectors in $\mathbb{R}^9$, by concatenation of either columns or rows).

For the *prolongation*, we set the values on the fine grid points as the average of the values of their neighbors on the coarse grid:

$$I^l_{i,j} = \text{mean}_{\substack{i' \in \text{rd}(i/2) \\ j' \in \text{rd}(j/2)}} \left(I^{l+1}_{i',j'} \right), \tag{4.2}$$

where

$$\text{rd}(k/2) = \begin{cases} \{k/2\} & k \text{ even}, \\ \{(k-1)/2, (k+1)/2\} & k \text{ odd}. \end{cases}$$

For example, to prolongate a 2×2 image to a 3×3, $\mathcal{P}$ is defined with

$$P^T = \frac{1}{4} \begin{pmatrix} 4 & 2 & 0 & 2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 2 & 4 & 0 & 1 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 1 & 0 & 4 & 2 & 0 \\ 0 & 0 & 0 & 0 & 1 & 2 & 0 & 2 & 4 \end{pmatrix}.$$

Note that these transformations clearly preserve nonnegativity.

4.3 Multigrid Cycle

Now that grids and transfer operators are defined, we need to choose the procedure that is applied at each grid level as it moves through the grid hierarchy. In this section, we propose three different approaches: nested iteration, V-cycle and full multigrid cycle.

In our settings, the transfer operators only change the number of rows m of the input matrix M, i.e., the number of pixels in the images of the database: the size of the images are approximatively four times smaller between each level: $m' \approx \frac{1}{4}m$. Since the computational complexity per iteration of the three algorithms (ANLS, MU and HALS) is almost proportional to m (cf. Appendix A), the iterations will be approximately four times cheaper. A possible way to allocate the time spent at each level is to allow the same number of iterations at each level, which seems to give good results in practice. Table 1 shows the time spent and the corresponding number of iterations performed at each level.

Table 1: Number of iterations performed and time spent at each level when allocating among L levels a total computational budget T corresponding to $4k$ iterations at the finest level.

	Level 1 (finer)	Level 2	... ...	Level $L-1$	Level L (coarser)	Total
$\sim$ # *iterations*	$3k$	$3k$	...	$3k$	$4k$	$(3L+1)k$
time	$\frac{3}{4}T$	$\frac{3}{16}T$	...	$\frac{3}{4^{L-1}}T$	$\frac{1}{4^{L-1}}T$	T

Note that the transfer operators require $O(mn)$ operations and since they are only performed once between each level, their computational cost can be neglected (at least for $r \gg 1$ and/or when a sizeable amount of iterations are performed).

4.3.1 Nested Iteration (NI)

To initialize NMF algorithms, we propose to factorize the image at the coarsest resolution and then use the solution as a initial guess for the next (finer) resolution. This is referred to as *nested iteration*, see Figure 4 for an illustration with three levels and Algorithm 4 for the implementation. The idea is to start off the final iterations at the finer level with a better initial estimate, thus reducing the

computational time required for the convergence of the iterative methods on the fine grid. The number of iterations and time spent at each level is chosen according to Table 1, i.e., three quarters of the alloted time for iterations at the current level preceded by one quarter of the time for the recursive call to the immediately coarser level.

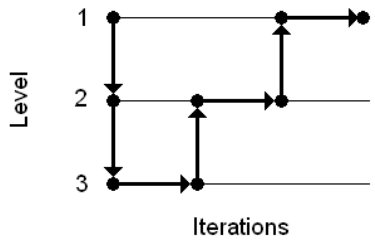


Figure 4: Nested Iteration. Transition between different levels for nested iteration.

Algorithm 4 Nested Iteration

Require: $L \in \mathbb{N}$ (number of levels), $M \in \mathbb{R}_+^{m\times n}$ (data matrix), $(V_0, W_0) \in \mathbb{R}_+^{m\times r} \times \mathbb{R}_+^{r\times n}$ (initial matrices) and $T \geq 0$ (total time allocated to the algorithm).

Ensure: $(V, W) \geq \mathbf{0}$ s.t. $VW \approx M$.

1: **if** $L = 1$ **then**
2: $[V, W] = \text{NMF algorithm}(M, V_0, W_0, T)$;
3: **else**
4: $M' = \mathcal{R}(M)$; $V_0' = \mathcal{R}(V_0)$;
5: $[V', W] = \text{Nested Iteration}(L-1, M', V_0', W_0, T/4)$;
6: $V = \mathcal{P}(V')$;
7: $[V, W] = \text{NMF algorithm}(M, V, W, 3T/4)$;
8: **end if**

Remark 1. *When the ANLS algorithm is used, the prolongation of V' does not need to be computed since that algorithm only needs an initial value for the W iterate. Note that this can be used in principle to avoid computing any prolongation, by setting V directly as the optimal solution of the corresponding NNLS problem.*

4.3.2 V–Cycle (VC)

A drawback of nested iteration is that it does not take advantage of the smoothing properties of iterations on fine grids (high-frequency components of the error are reduced faster). It is therefore often more efficient to perform a few iterations at the fine level before going to coarser levels. The simplest choice is referred to as V-cycle and is illustrated on Figure 5 with three levels; see Algorithm 5 for the implementation. Time allocation is as follows: one quarter of the alloted time is devoted to iterations at the current level, followed by one quarter of the time for the recursive call to the immediately coarser level, and finally one half of the time again for iterations at the current level (we have therefore three quarters of the total time spent for iterations at current level, as for nested iteration).

4.3.3 Full Multigrid (FMG)

Combining ideas of nested iteration and V-cycle leads to a full multigrid cycle defined recursively as follows: at each level, a V-cycle is initialized with the solution obtained at the underlying level using

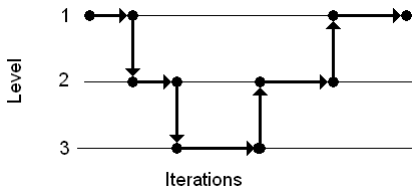


Figure 5: V-cycle. Transition between different levels for V-cycle.

Algorithm 5 V-cycle

Require: $L \in \mathbb{N}$ (number of levels), $M \in \mathbb{R}_+^{m \times n}$ (data matrix), $(V_0, W_0) \in \mathbb{R}_+^{m \times r} \times \mathbb{R}_+^{r \times n}$ (initial matrices) and $T \geq 0$ (total time allocated to the algorithm).

Ensure: $(V, W) \geq \mathbf{0}$ s.t. $VW \approx M$.

1: **if** $L = 1$ **then**
2: $[V, W] = \text{NMF algorithm}(M, V_0, W_0, T)$;
3: **else**
4: $[V, W] = \text{NMF algorithm}(M, V_0, W_0, T/4)$;
5: $M' = \mathcal{R}(M)$; $V' = \mathcal{R}(V)$;
6: $[V', W] = \text{V-cycle}(L - 1, M', V', W, T/4)$;
7: $V = \mathcal{P}(V')$;
8: $[V, W] = \text{NMF algorithm}(M, V, W, T/2)$;
9: **end if**

a full-multigrid cycle. This is typically the most efficient multigrid strategy [31]. In this case, we propose to partition the time as follows (T is the total time): $\frac{T}{4}$ for the initialization (call of the full multigrid on the underlying level) and $\frac{3T}{4}$ for the V-cycle at the current level (cf. Algorithm 6).

Algorithm 6 Full Multigrid

Require: $L \in \mathbb{N}$ (number of levels), $M \in \mathbb{R}_+^{m \times n}$ (data matrix), $(V_0, W_0) \in \mathbb{R}_+^{m \times r} \times \mathbb{R}_+^{r \times n}$ (initial matrices) and $T \geq 0$ (total time allocated to the algorithm).

Ensure: $(V, W) \geq \mathbf{0}$ s.t. $VW \approx M$.

1: **if** $L = 1$ **then**
2: $[V, W] = \text{NMF algorithm}(M, V_0, W_0, T)$;
3: **else**
4: $V' = \mathcal{R}(V_0)$; $M' = \mathcal{R}(M)$; *
5: $[V', W] = \text{Full Multigrid}(L - 1, M', V', W_0, T/4)$;
6: $V = \text{prolongation}(V')$;
7: $[V, W] = \text{V-cycle}(L, M, V, W, 3T/4)$;
8: **end if**

*Note that the restrictions of M should be computed only once for each level and saved as global variables so that the call of the V-cycle (step 7) does not have to recompute them.

4.4 Smoothing Properties

We explained why the multilevel strategy was potentially able to accelerate iterative algorithms for NMF: cheaper computations and smoothing of the error on coarse levels. Before giving extensive numerical results in Section 5, we illustrate this crucial feature of multilevel methods on the ORL face

database.

Comparing three levels, Figure 6 displays the error (after prolongation to the fine level) for two faces and for different number of iterations (10, 50 and 100) using MU. Comparing the first row and

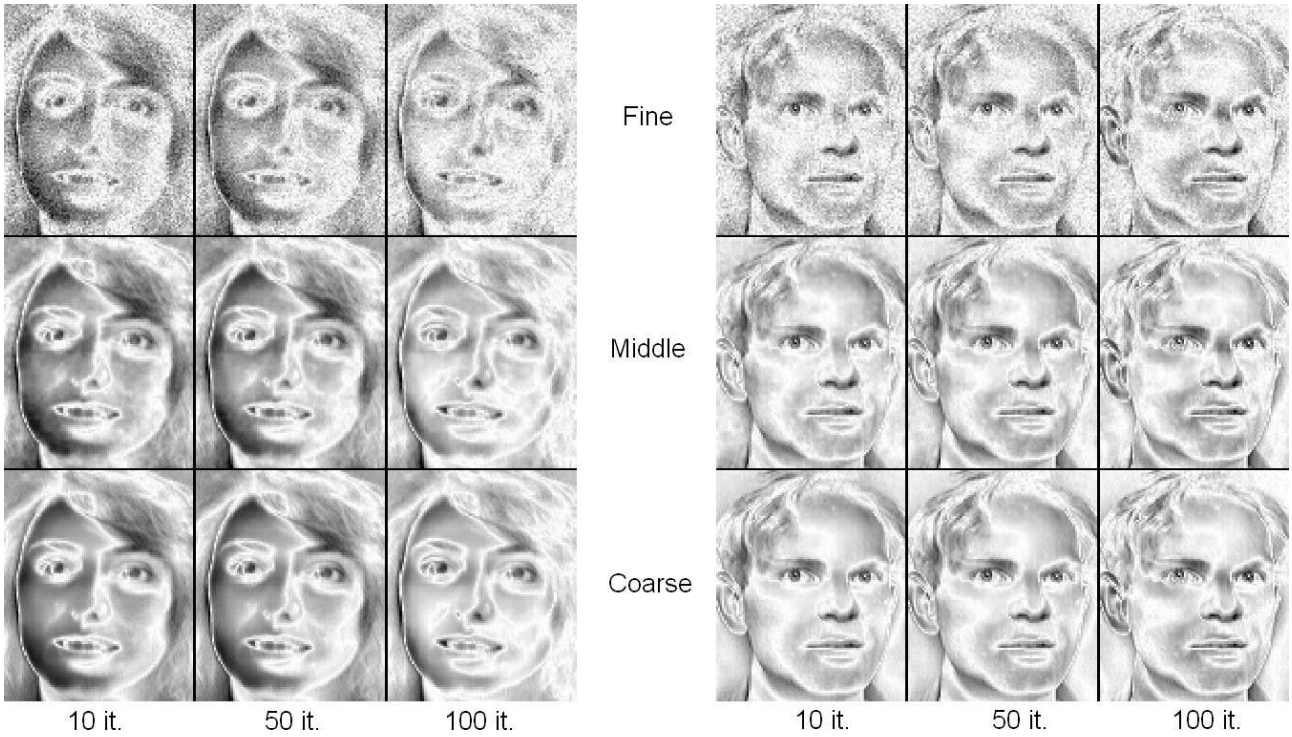


Figure 6: Smoothing on Coarse Levels. Example of the smoothing properties of the multilevel approach on the ORL face database. Each image represents the absolute value of the approximation error (black tones indicate a high error) of one of the two faces from the ORL face database. These approximations are the prolongations (to the fine level) of the solutions obtained using the multiplicative updates on a single level, with $r = 40$ and the same initial matrices. From top to bottom: level 1 (fine), level 2 (middle) and level 3 (coarse); from left to right: 10 iterations, 50 iterations and 100 iterations.

the last row of Figure 6, it is clear that, in this example, the multilevel approach allows a significant smoothing of the error. Already after 10 iterations, the error obtained with the prolongated solution of the coarse level is smoother and smaller (see Figure 7) while it is computed much faster.

Figure 7 gives the evolution of the error with respect to the number of iterations performed (left) and with respect to computational time (right). In this example, the initial convergence on the three levels is comparable, while the computational cost is much cheaper on coarse levels. In fact, compared to the fine level, the middle (resp. coarse) level is approximately 4 (resp. 16) times cheaper.

5 Computational Results

To evaluate the performances of our multilevel approach, we present some numerical results for several standard image databases, see Table 2.

For each database, the multilevel strategy is tested using 100 runs initialized with the same random matrices for the three algorithms (ANLS, MU and HALS) and the three multigrid cycles (NI, V-cycle and FMG), with a time limit of 10 seconds. All algorithms have been implemented in MATLAB® 7.1 (R14) and tested on a 3GHz Intel® Core™2 Dual CPU PC.

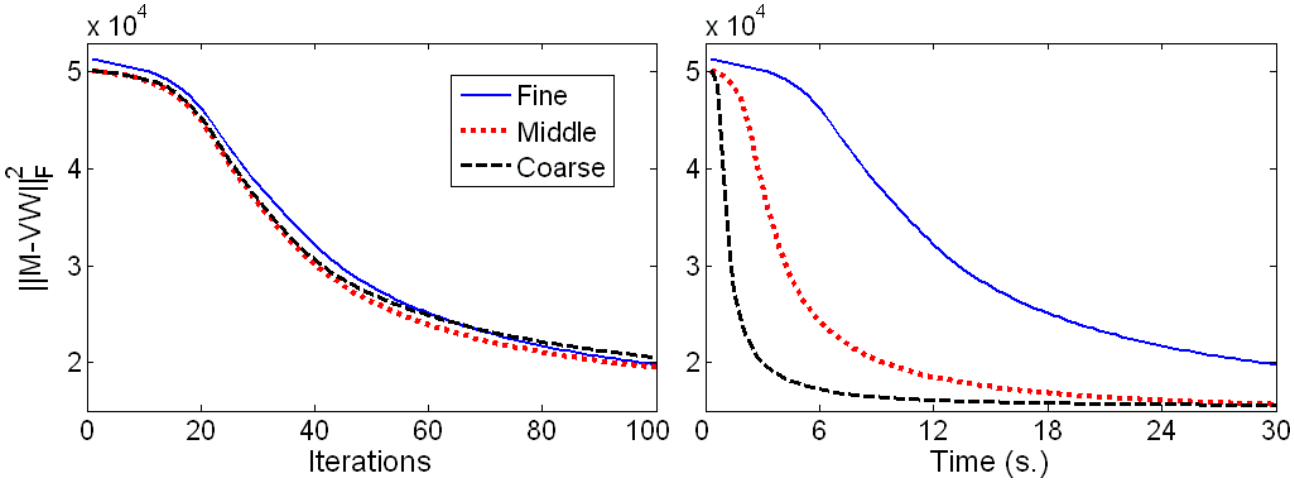


Figure 7: Evolution of the error on each level, after prolongation on the fine level, with respect to (left) the number of iterations performed and (right) the computational time. Same setting as in Figure 6.

Table 2: Image datasets.

Data	# pixels	m	n	r
ORL face[1]	112×92	10304	400	40
Umist face[2]	112×92	10304	575	40
Iris[3]	960×1280	1228800	8	4
Hubble Telescope [34]	128×128	16384	100	8

[1] *http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html*
[2] *http://www.cs.toronto.edu/~roweis/data.html*
[3] *http://www.bath.ac.uk/elec-eng/research/sipg*

5.1 Results

Tables 3, 4 and 5 give the mean error attained within 10 seconds using the different approaches.

Table 3: Comparison of the mean error on the 100 runs with ANLS.

	# lvl	ORL	Umist	Iris	Hubble
NMF	1	14960	26013	28934	24.35
NI	2	14683	25060	27834	**15.94**
	3	14591	24887	27572	16.93
	4	14580	24923	27453	17.20
VC	2	14696	25195	27957	16.00
	3	14610	24848	27620	16.12
	4	14599	24962	27490	16.10
FMG	2	14683	25060	27821	16.10
	3	14516	24672	27500	16.56
	4	**14460**	**24393**	**27359**	16.70

Table 4: Comparison of the mean error on the 100 runs with MU.

	# lvl	ORL	Umist	Iris	Hubble
NMF	1	34733	131087	64046	21.68
NI	2	23422	87966	37604	22.80
	3	20502	67131	33114	18.49
	4	19507	59879	31146	16.19
VC	2	23490	90064	36545	10.62
	3	20678	69208	32086	9.77
	4	19804	62420	30415	**9.36**
FMG	2	23422	87966	37504	22.91
	3	19170	58469	32120	15.06
	4	**17635**	**46570**	**29659**	11.71

Table 5: Comparison of the mean error on the 100 runs with HALS.

	# lvl	ORL	Umist	Iris	Hubble
NMF	1	15096	27544	31571	17.97
NI	2	14517	25153	29032	17.37
	3	14310	24427	28131	16.91
	4	14280	24256	27744	16.92
VC	2	14523	25123	28732	17.37
	3	14339	24459	28001	17.02
	4	14327	24364	27670	17.04
FMG	2	14518	25153	29120	17.39
	3	14204	23950	27933	**16.69**
	4	**14107**	**23533**	**27538**	16.89

In all the cases, the multilevel approaches generate much better solutions than the original NMF algorithms; indicating that it is able to accelerate their convergence. The full multigrid cycle is, as expected, the best strategy while nested iteration and V-cycle give comparable performances. We also observe that the additional speed up of the convergence when the number of levels is increased from 3 to 4 is less significant; it is even slightly reduced in some cases. In general, the 'optimal' number of levels will depend on the size and the smoothness of the data.

HALS combined with the the full multigrid cycle is one of the best strategies. Figure 8 displays the distribution of the errors for the different databases in this particular case. For the ORL and Umist databases, the multilevel strategy is extremely efficient: all the solutions generated with 2 and 3 levels are better than the original NMF algorithm. For the Iris and Hubble databases, the difference is not as clear. The reason is that the corresponding NMF problems are 'easier' because the rank r is smaller. Hence the algorithms converge faster to stationary points, and the distribution of the final errors is more concentrated.

In order to visualize the evolution of the error through the iterations, Figure 9 displays the evolution of the objective function with respect to the number of iterations independently for each algorithm and each database using nested iteration as the multigrid cycle (which is the easiest to represent). In all the cases, the prolongations of the solutions from the lower levels generate much better solutions that the one obtained on the fine level.

These test results are very encouraging: the multilevel approach for NMF seems very efficient and allows to speed up convergence of algorithms significantly.

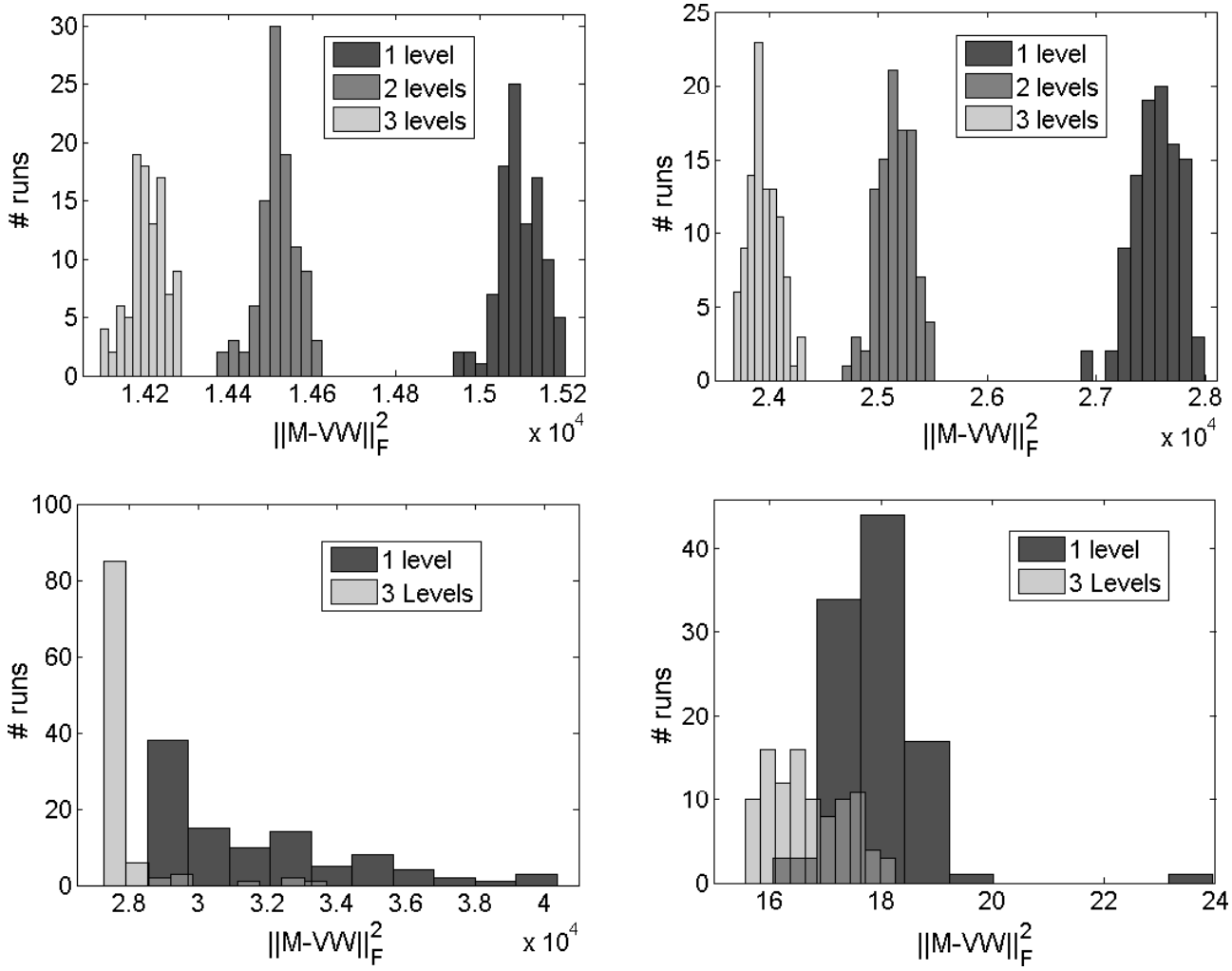


Figure 8: Distribution of the error among the 100 random initializations using the HALS algorithm with a full multigrid cycle: (top left) ORL, (top right) Umist, (bottom left) Iris, and (bottom right) Hubble.

6 Concluding Remarks

In this paper, a multilevel approach to speed up NMF algorithms has been proposed and its efficiency has been experimentally demonstrated. In order to use this technique, one needs to be able to design linear operators preserving nonnegativity and transferring accurately the data between the different levels. To conclude, we give some directions for further research.

6.1 Extensions

We have only used our multilevel approach for a specific objective function (sum of squared errors) to speed up three NMF algorithms (ANLS, MU and HALS) and to factorize 2D images. However, this can be easily generalized to any other objective function, any other iterative algorithm and applied to other kind of smooth data. Moreover, other types of coarse grid definition, transfer operators and grid cycle can be used and could potentially improve efficiency.

A limitation of the proposed approach is that the multigrid strategy is only applied to one dimension of the matrix (because we did not assume that the different images are related to each other in any

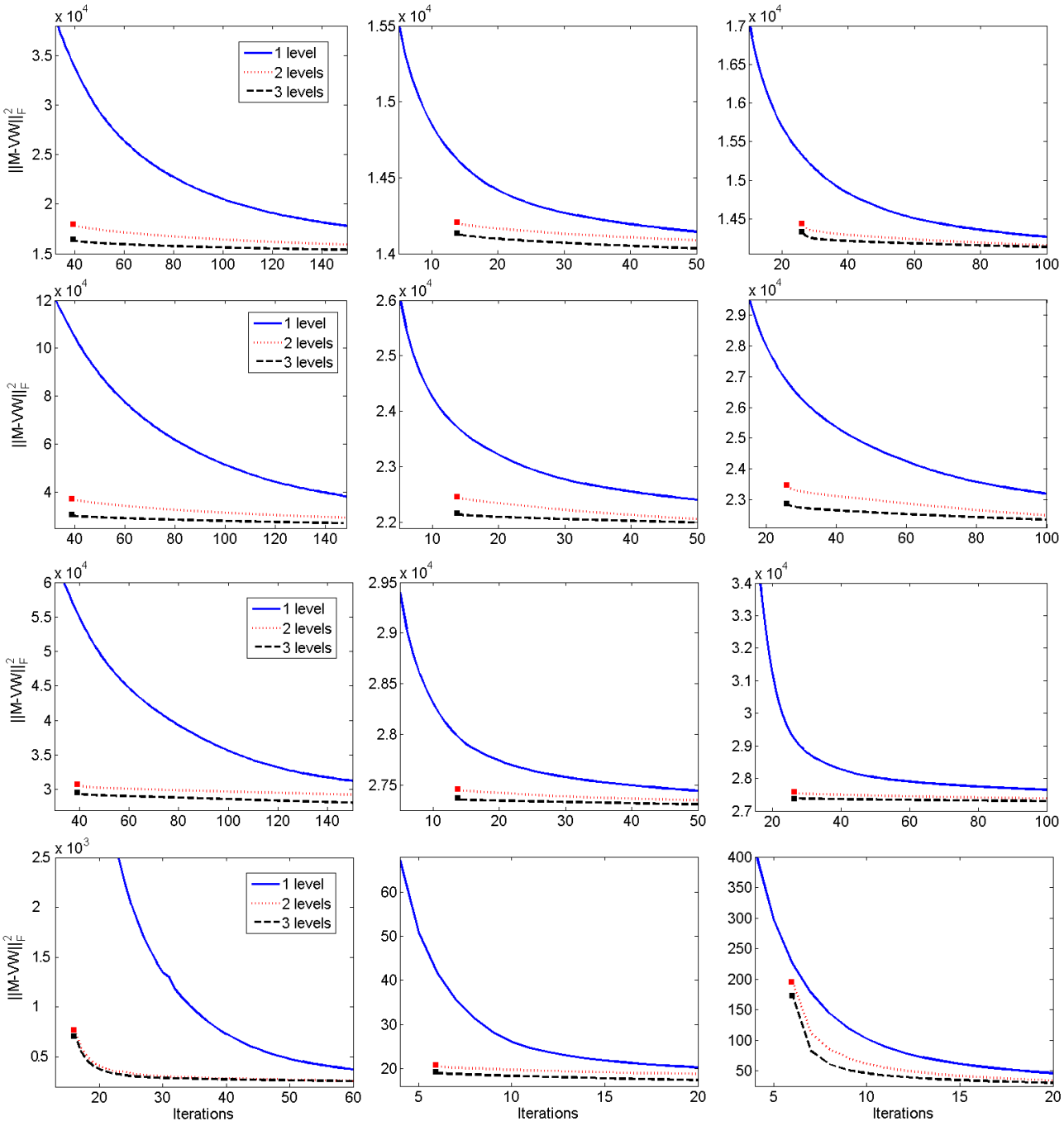


Figure 9: Evolution of the objective function. From left to right : MU, ANLS and HALS. From top to bottom: ORL, Umist, Iris and Hubble databases. *1 level* stands for the standard NMF algorithms. The initial points for the curves *2 levels* and *3 levels* are the prolongated solutions obtained on the coarser levels using nested iteration, cf. Section 4.3. All algorithms were initialized with the same random matrices.

way). However, in some applications, rows of matrix M might also be restricted to lower dimensional spaces. For example, in hyperspectral data analysis, each column of matrix M represents an image at a given wavelength, while each row represents the spectral signature of a pixel, see, e.g., [28, 17]. Since spectral signatures feature smooth components as well, the multilevel strategy can be used to

reduce both dimensions of the data matrix.

This idea can also be extended to Nonnegative Tensor Factorization (NTF) (see, e.g., [34] and references therein where it is used to analyze the hyperspectral Hubble telescope images) by using multilevel techniques for higher dimensional spaces.

6.2 Initialization

Several judicious initializations for NMF algorithms have been proposed in the literature and allow to speed up convergence and improve, in general, the final solution [12, 2]. The computational cost of these good initial guesses depends on the matrix dimensions and will then be cheaper to compute on the coarsest grid. Therefore, it would be interesting to combine classical NMF initializations techniques with our multilevel approach for further speed up.

6.3 Unstructured data

A priori, applying a multilevel method to data for which we do not have any information about the matrix to factorize (and a fortiori about the solution) seems out of reach. In fact, in these circumstances, there is no sensible way to define the transfer operators.

However, it is not hopeless to extend the multilevel idea to other type of data. For example, in text mining applications, the term-by-document matrix could be restricted by stacking synonyms or similar texts together (similarly as in [29]). Of course, this implies some a priori knowledge or preprocessing of the data (which should be cheap enough to be profitable).

Acknowledgments

We thank Quentin Rentmeesters for his helpful comments.

References

[1] M. Berry, M. Browne, A. Langville, P. Pauca, and R. Plemmons, *Algorithms and Applications for Approximate Nonnegative Matrix Factorization*, Computational Statistics and Data Analysis, 52 (2007), pp. 155–173.

[2] C. Boutsidis and E. Gallopoulos, *SVD based initialization: A head start for nonnegative matrix factorization*, Journal of Pattern Recognition, 41 (2008), pp. 1350–1362.

[3] J. H. Bramble, *Multigrid methods*, Number 294 Pitman Research Notes in Mathematic Series. Longman Scientific & Technical, UK, 1995.

[4] A. Brandt, *Guide to multigrid development*, W. Hackbusch and U. Trottenberg, eds., Multigrid Methods, Lecture Notes in Mathematics, Springer, 960 (1982), pp. 220–312.

[5] W. L. Briggs, *A Multigrid Tutorial*, SIAM, Philadelphia, 1987.

[6] D. Chen and R. Plemmons, *Nonnegativity Constraints in Numerical Analysis*, in A. Bultheel and R. Cools (Eds.), Symposium on the Birth of Numerical Analysis, World Scientific Press., 2009.

[7] A. Cichocki, S. Amari, R. Zdunek, and A. Phan, *Non-negative Matrix and Tensor Factorizations: Applications to Exploratory Multi-way Data Analysis and Blind Source Separation*, Wiley-Blackwell, 2009.

[8] C. Cichocki and A.-H. Phan, *Fast local algorithms for large scale Nonnegative Matrix and Tensor Factorizations*, IEICE Transactions on Fundamentals of Electronics, Vol. E92-A No.3 (2009), pp. 708–721.

[9] C. Cichocki, R. Zdunek, and S. Amari, *Non-negative Matrix Factorization with Quasi-Newton Optimization*, in Lecture Notes in Artificial Intelligence, Springer, vol. 4029, 2006, pp. 870–879.

[10] ——, *Hierarchical ALS Algorithms for Nonnegative Matrix and 3D Tensor Factorization*, in Lecture Notes in Computer Science, Vol. 4666, Springer, pp. 169-176, 2007.

[11] ——, *Nonnegative Matrix and Tensor Factorization*, IEEE Signal Processing Magazine, (2008), pp. 142–145.

[12] J. Curry, A. Dougherty, and S. Wild, *Improving non-negative matrix factorizations through structured initialization*, Journal of Pattern Recognition, 37(11) (2004), pp. 2217–2232.

[13] M. E. Daube-Witherspoon and G. Muehllehner, *An iterative image space reconstruction algorithm suitable for volume ect*, IEEE Trans. Med. Imaging, 5 (1986), pp. 61–66.

[14] I. Dhillon, D. Kim, and S. Sra, *Fast Newton-type Methods for the Least Squares Nonnegative Matrix Approximation problem*, in Proc. of SIAM Conf. on Data Mining, 2007.

[15] N. Gillis and F. Glineur, *Nonnegative Factorization and The Maximum Edge Biclique Problem*. CORE Discussion paper 2008/64, 2008.

[16] ——, *Nonnegative Matrix Factorization and Underapproximation*. Communication at 9th International Symposium on Iterative Methods in Scientific Computing, Lille, France, 2008.

[17] N. Gillis and R. Plemmons, *Dimensionality reduction, classification, and spectral mixture analysis using nonnegative underapproximation*, in SPIE conference Volume 7695, paper 46, Orlando, 2010.

[18] S. Gratton, A. Sartenaer, and P. Toint, *On Recursive Multiscale Trust-Region Algorithms for Unconstrained Minimization*, in Oberwolfach Reports: Optimization and Applications.

[19] J. Han, L. Han, M. Neumann, and U. Prasad, *On the rate of convergence of the image space reconstruction algorithm*, Operators and Matrices, 3(1) (2009), pp. 41–58.

[20] N.-D. Ho, P. Van Dooren, and V. Blondel, *Descent methods for nonnegative matrix factorization*, In: Numerical Linear Algebra in Signals, Systems and Control, Springer Verlag, (2008).

[21] H. Kim and H. Park, *Non-negative Matrix Factorization Based on Alternating Non-negativity Constrained Least Squares and Active Set Method*, SIAM J. Matrix Anal. Appl., 30(2) (2008), pp. 713–730.

[22] C. Lawson and R. Hanson, *Solving Least Squares Problems*, Prentice-Hall, 1974.

[23] D. Lee and H. Seung, *Learning the Parts of Objects by Nonnegative Matrix Factorization*, Nature, 401 (1999), pp. 788–791.

[24] ——, *Algorithms for Non-negative Matrix Factorization*, In Advances in Neural Information Processing, 13 (2001).

[25] C.-J. Lin, *On the Convergence of Multiplicative Update Algorithms for Nonnegative Matrix Factorization*, in IEEE Transactions on Neural Networks, 2007.

[26] ——, *Projected Gradient Methods for Nonnegative Matrix Factorization*, Neural Computation, 19 (2007), pp. 2756–2779. MIT press.

[27] P. Paatero and U. Tapper, *Positive matrix factorization: a non-negative factor model with optimal utilization of error estimates of data values*, Environmetrics, 5 (1994), pp. 111–126.

[28] P. Pauca, J. Piper, and R. Plemmons, *Nonnegative matrix factorization for spectral data analysis* , Linear Algebra and its Applications, 406(1) (2006), pp. 29–47.

[29] S. Sakellaridi, H.-r. Fang, and Y. Saad, *Graph-based Multilevel Dimensionality Reduction with Applications to Eigenfaces and Latent Semantic Indexing.* preprint, 2009.

[30] D. Terzopoulos, *Image Analysis Using Multigrid Relaxation Methods*, J. Math. Phys., PAMI-8(2) (1986), pp. 129–139.

[31] U. Trottenberg, C. Oosterlee, and A. Schüller, *Multigrid*, Elsevier Academic Press, London, 2001.

[32] M. Van Benthem and M. Keenan, *Fast algorithm for the solution of large-scale non-negativity constrained least squares problems*, J. Chemometrics, 18 (2004), pp. 441–450.

[33] S. A. Vavasis, *On the complexity of nonnegative matrix factorization*, SIAM Journal on Optimization, 20 (2009), pp. 1364–1377.

[34] Q. Zhang, H. Wang, R. Plemmons, and P. Pauca, *Tensor methods for hyperspectral data analysis: a space object material identification study*, J. Optical Soc. Amer. A, 25(12) (2008), pp. 3001–3012.

A Computational Cost of ANLS, MU and HALS

A.1 Active Set Methods for NNLS

In a nutshell, active set methods for nonnegative least squares work in the following iterative fashion [22]

1. get rid of the nonnegativity constraints and solve the unconstrained least squares problem corresponding to the set of passive (nonzero) variables;
2. check the optimality conditions, if they are not satisfied:
3. update the sets of active (zero) and passive (nonzero) variables accordingly;

in such a way that that the objective function is decreased at each step.

In (NMF), the problem of computing the optimal V (resp. W) for a given fixed W (resp. V) can be decoupled into m (resp. n) independent NNLS subproblems in r variables:

$$\min_{V_{i:}\in\mathbb{R}^r_+} ||M_{i:} - V_{i:}W||_F^2,\ 1 \le i \le m \quad (\text{resp. } \min_{W_{:j}\in\mathbb{R}^r_+} ||M_{:j} - VW_{:j}||_F^2,\ 1 \le j \le n).$$

Each of them amounts to solving a sequence of linear subsystems (with at most r variables, cf. step 1 above) of

$$V_{i:}(WW^T) = M_{i:}W^T,\ 1 \le i \le m \quad (\text{resp. } (V^TV)W_{:j} = V^TM_{:j},\ 1 \le j \le n).$$

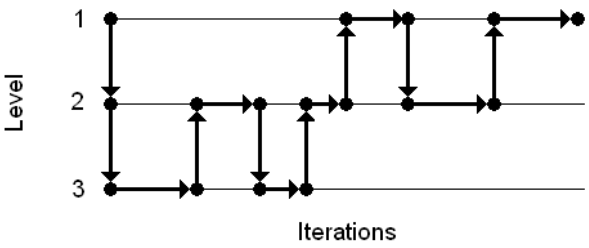
In the worst case, one might have to solve every possible subsystem, which requires $O(g(r))$ operations with[4] $g(r) = \sum_{i=1}^r \binom{r}{i} i^3 = \Theta(2^r r^3)$. Note that WW^T and MW^T (resp. V^TV and V^TM) can be computed once for all to avoid redundant computations. Finally, one ANLS step requires at most $O(mnr + (m+n)s(r)r^3)$ operations per iteration, where $s(r) \le 2^r$. In the worst case, $s(r)$ is in $O(2^r)$ while in practice it is in general much lower (as is the case for the simplex method for linear programming).

When m is reduced by a certain factor (e.g., 4 as in our experimental results), the computational cost is not exactly reduced by the same factor because of the $(m+n)$ factor above. However, in our applications, m (number of pixels) is much larger than n (number of images) and therefore one can approximate the cost per iteration to be reduced by the same factor since $\frac{m+n}{4} \approx \frac{m}{4}$.

A.2 MU and HALS

The main computational cost of both MU and HALS resides in the computation of MW^T, V^TM, WW^T and V^TV which requires $4mnr + 4(m+n)r^2 = O(mnr)$ operations, cf. Algorithms 2 and 3. The computational cost is almost proportional to m (only the nr^2 term is not, which negligible for $m \gg r$).

[4] One can check that $(2^{(r-3)} - 1)(r-2)^3 \le g(r) \le 2^r r^3$.


1
2
3
Level
Iterations